US006071317A

United States Patent [19]

Nagel

[11] **Patent Number: 6,071,317**

[45] **Date of Patent: Jun. 6, 2000**

[54] **OBJECT CODE LOGIC ANALYSIS AND AUTOMATED MODIFICATION SYSTEM AND METHOD**

[75] Inventor: **Robert H. Nagel**, New York, N.Y.

[73] Assignee: **Digits Corp.**, N.Y.

[21] Appl. No.: **09/208,148**

[22] Filed: **Dec. 9, 1998**

Related U.S. Application Data

[60] Provisional application No. 60/069,211, Dec. 11, 1997.

[51] **Int. Cl.**^{7} **G06F 9/445**

[52] **U.S. Cl.** **717/4**; 717/1; 717/2; 717/3; 717/5; 717/6; 717/7; 717/8; 717/9; 717/10; 717/11

[58] **Field of Search** 395/701, 702, 395/703, 704, 705, 706, 707; 717/1, 2, 3, 4, 5, 6, 7, 8, 9, 11

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,954,941	9/1990	Redman	364/200
5,175,828	12/1992	Hall et al.	395/375
5,359,730	10/1994	Marron	395/650
5,459,866	10/1995	Akiba et al.	395/650
5,481,713	1/1996	Wetmore et al.	395/703
5,495,612	2/1996	Hirayama et al.	395/700
5,507,030	4/1996	Sites	395/800
5,555,418	9/1996	Nilsson et al.	395/700
5,600,836	2/1997	Alter	395/612
5,630,118	5/1997	Shaughnessy	707/1
5,644,762	7/1997	Soeder	707/6

(List continued on next page.)

OTHER PUBLICATIONS

"Storing All Dates in Character Format", IBM Technical Disclosure Bulletin, Jul. 1991, 83–84.

IBM Technical Disclosure Bullentin vol. 312 No. 1 Dual Indirect RAM/ROM Jump Tables for Firmware Updates, Jun. 1988.

IBM Technical Disclosure Bulletin vol. 27 No. 4a Method of Customizing Patches For Each Hardware Configuration, Sep. 1984.

IBM Technical Disclosure Bulletin vol. 35 No. 7 Method and Mechanism For Dynamic Loader, Dec. 1992.

Code Complete Steve McConnell pp. 379–380, 1993.

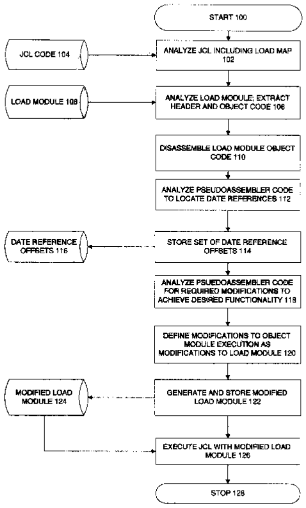
Fuzzt–Neural Control Junhong and Linkens pp. 1–9 and 34–36, 1995.

Primary Examiner—Tariq R. Hafiz
Assistant Examiner—Ted T. Vo
Attorney, Agent, or Firm—Milde, Hoffberg & Macklin, LLP

[57] **ABSTRACT**

A method and system for modifying computer program logic with respect to a predetermined aspect, comprising (a) before run time: analyzing compiled computer program logic of a module for processes involving the predetermined aspect before run time, substantially without decompilation or reference to computer program source code; and storing a set of modifications relating to computer program logic modifications of the module relating to the predetermined aspect; and (b) at run time: based on the stored set of modifications, selectively transferring program control from the module to a separate logical structure, executing modified logical operations with respect to the predetermined aspect, and subsequently returning program control to the module. The predetermined aspect may be, for example, a data type, algorithm type, or interface specification. In a preferred embodiment, the predetermined aspect is date related data, and more particularly, to logical operations relating to date related data which are flawed. The system preferably operates in a mainframe environment, wherein the compiled computer program constitutes one or more load modules, executing under an operating system, wherein the computer program logic modifications preferably comprise program flow control diversions in an original object module, which selectively transfer logical control to a separate object module to effect modifications to the computer program logic, followed by a return of control to the original object module.

31 Claims, 1 Drawing Sheet


START 100
JCL CODE 104
ANALYZE JCL INCLUDING LOAD MAP 102
LOAD MODULE 108
ANALYZE LOAD MODULE; EXTRACT HEADER AND OBJECT CODE 106
DISASSEMBLE LOAD MODULE OBJECT CODE 110
ANALYZE PSEUDOASSEMBLER CODE TO LOCATE DATE REFERENCES 112
DATE REFERENCE OFFSETS 116
STORE SET OF DATE REFERENCE OFFSETS 114
ANALYZE PSUEDOASSEMBLER CODE FOR REQUIRED MODIFICATIONS TO ACHIEVE DESIRED FUNCTIONALITY 118
DEFINE MODIFICATIONS TO OBJECT MODULE EXECUTION AS MODIFICATIONS TO LOAD MODULE 120
MODIFIED LOAD MODULE 124
GENERATE AND STORE MODIFIED LOAD MODULE 122
EXECUTE JCL WITH MODIFIED LOAD MODULE 126
STOP 128